NHR FAU

FAU Friedrich-Alexander-Universität Erlangen-Nürnberg

Programming Techniques for Supercomputers Tutorial

Erlangen National High Performance Computing Center

Department of Computer Science

FAU Erlangen-Nürnberg

Sommersemester 2026

Assignment 6 – Task 1

```
$ likwid-topology -g
--------------------------------------------------------------------------------
CPU name:       Intel(R) Xeon(R) Platinum 8470
CPU type:       Intel SapphireRapids processor
CPU stepping:   8
********************************************************************************
Hardware Thread Topology
********************************************************************************
Sockets:                2
CPU dies:               2
Cores per socket:       52
Threads per core:       1
--------------------------------------------------------------------------------
HWThread        Thread        Core        Die        Socket        Available
0               0             0           0          0             *
1               0             1           0          0             *
2               0             2           0          0             *
...
101             0             101         0          1             *
102             0             102         0          1             *
103             0             103         0          1             *
--------------------------------------------------------------------------------
Socket 0:               ( 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 )
Socket 1:               ( 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 )
```

2 sockets,
104 cores/node

Assignment 6 – Task 1

```
$ likwid-topology -g
--------------------------------------------------------------------------------
Cache Topology
--------------------------------------------------------------------------------
Level:                  1
Size:                   48 kB
Cache groups:           ( 0 ) ( 1 ) ( 2 ) […] ( 103 )
--------------------------------------------------------------------------------
Level:                  2
Size:                   2 MB
Cache groups:           ( 0 ) ( 1 ) ( 2 ) […] ( 103 )
--------------------------------------------------------------------------------
Level:                  3
Size:                   105 MB
Cache groups:           ( 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 ) ( 52 53 54 55
56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86
87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 )
--------------------------------------------------------------------------------
```

Per-core L1 (48 KiB) and L2 (2048 KiB) caches

chip-shared L3 (105 MiB) cache

Assignment 6 – Task 1

```
$ likwid-topology -g
********************************************************************************
NUMA Topology
********************************************************************************
NUMA domains:           8
--------------------------------------------------------------------------------
Domain:                 0
Processors:             ( 0 1 2 3 4 5 6 7 8 9 10 11 12 )
Distances:              10 12 12 12 21 21 21 21
Free memory:            257077 MB
Total memory:           257636 MB
--------------------------------------------------------------------------------
Domain:                 1
Processors:             ( 13 14 15 16 17 18 19 20 21 22 23 24 25 )
Distances:              12 10 12 12 21 21 21 21
Free memory:            257815 MB
Total memory:           258044 MB
[…]
```

8 NUMA domains,
4 per chip

Assignment 6 – Task 2

```
double t_start,t_end;
int k, NITER, N;
double *a = (double*)aligned_alloc(64,N*sizeof(double));
double *b = (double*)aligned_alloc(64,N*sizeof(double));

#pragma omp parallel for
for(k=0; k<N; ++k)
  a[k]=b[k]=0.;

NITER=1;
do {
  t_start = getTimeStamp();

  for(k=0; k<NITER; ++k) {
    #pragma omp parallel for
    for(i=0; i<N; ++i) {
      a[i] = a[i] + b[i];
    }
    if(a[N/2]<0.) printf("%lf",a[N/2]);
  }
  t_end = getTimeStamp();
  NITER = NITER*2;
} while (wct_end-wct_start<1.);

NITER = NITER/2;

printf("Total walltime: %f, NITER: %d, Perf: %.3lf GF/s\n", t_end-
t_start, NITER, 1.*(double)N*NITER/(t_end-t_start)*1.e-9);
```

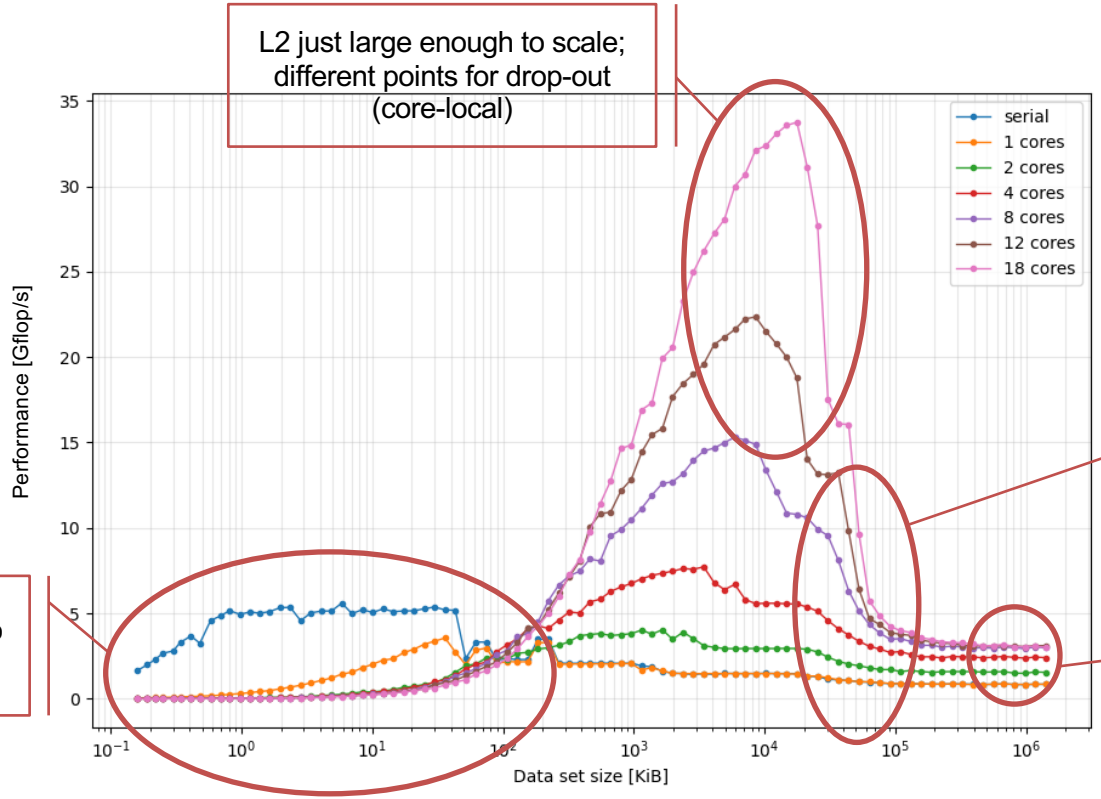
```
icx -O3 -xHost -qopenmp
 accumulate.c timing.c
```

Assignment 6 – Task 2

```
#/bin/bash
for t in 1 2 4 8 12 18; do
    n=10
    for i in `seq 1 90`; do
        echo -n $n " "
        srun -cpu-freq=2000000-2000000:performance likwid-pin -c 0-$(( t - 1 )) -q \
                                        ./a.out $n >> acc_${t}.out
        n=$(( (n * 12) / 10 ))
    done
done
```

```
#/bin/bash
for t in 1 2 4 8 12 18; do
    n=10
    for i in `seq 1 90`; do
        echo -n $n " "
        OMP_NUM_THREADS=$t OMP_PLACES=cores OMP_PROC_BIND=close \
                                srun -cpu-freq=2000000-2000000:performance \
                                ./a.out $n >> acc_${t}.out
        n=$(( (n * 12) / 10 ))
    done
done
```

Assignment 6 – Task 2 a)

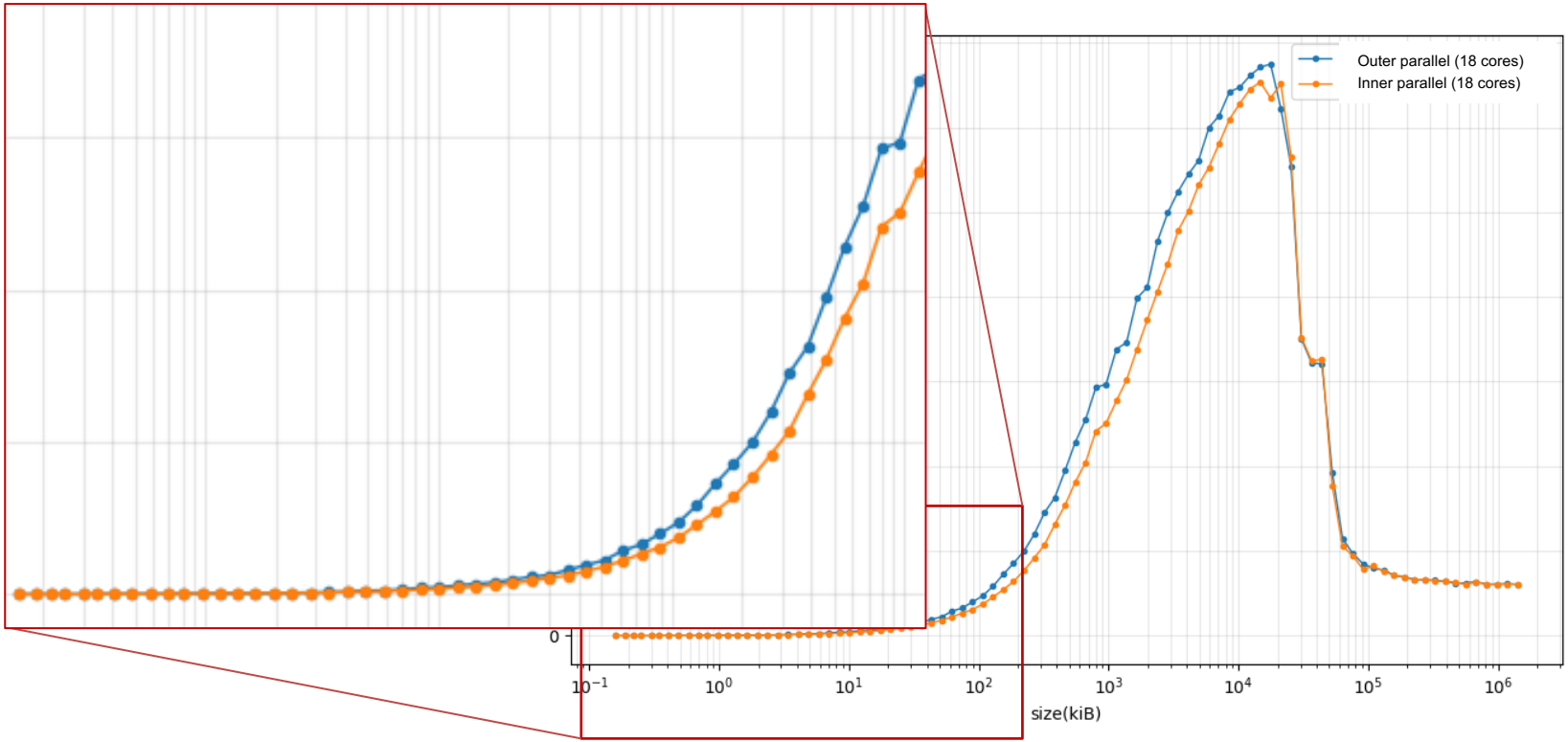


"Reverse" scaling due to OpenMP overhead

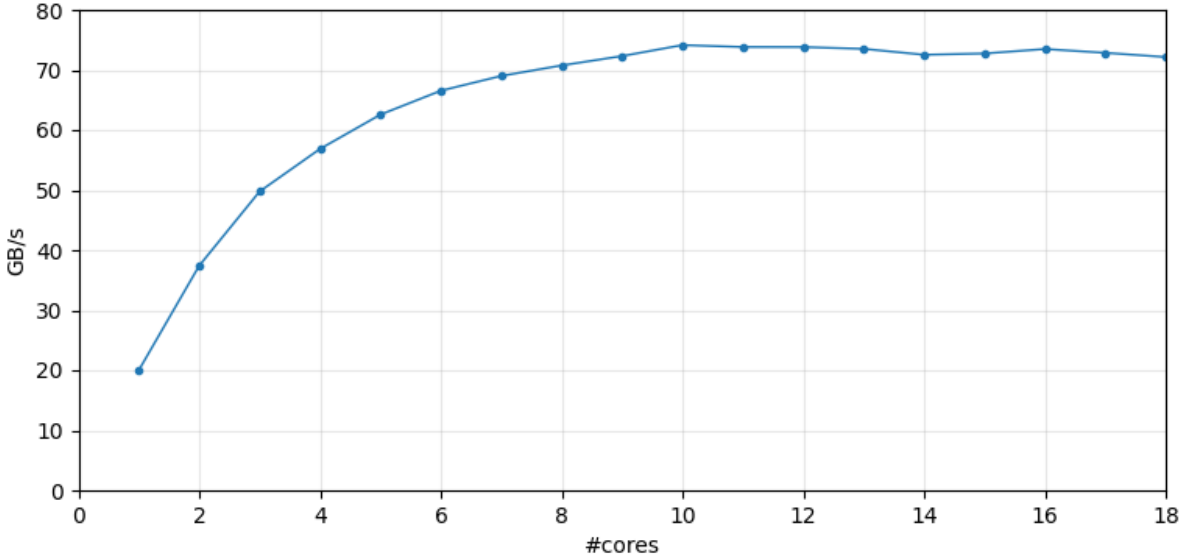
Decent scaling in L3 up to 8 cores; however, bad scaling onwards ➔ shared L3 cache

No scaling beyond 8 cores ➔ memory bandwidth exhausted

Assignment 6 – Task 2 b)



Assignment 6 – Task 2 c)

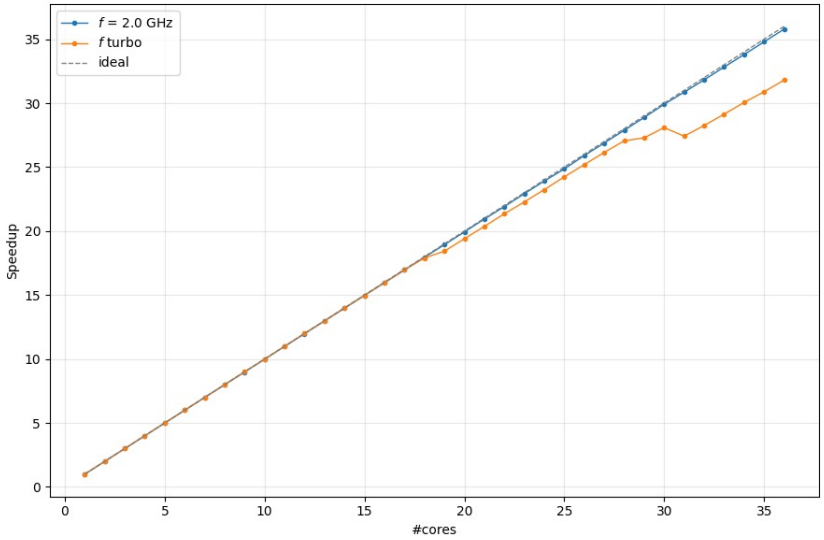



```
#pragma omp parallel for
for (int i=0; i<N; ++i)
    a[i] = a[i] + b[i];
```

$$b = 3\frac{GFLOP}{s} \times 24\ \frac{B}{FLOP} = 72\ \frac{GB}{s}$$

Speedup: ~ 3.6x

Assignment 6 – Task 3




```
// bogus parallel region
#pragma omp parallel
  if(omp_get_thread_num()==0) printf("Hello");

  S = getTimeStamp();
#pragma omp parallel for reduction(+:sum)
private(x)
  for(int i=0; i<N; i++) {
    x = (i + 0.5) * delta_x;
    sum = sum + 4.0 * sqrt(1 - x*x);
  }
  double result = sum * delta_x;
  E = getTimeStamp();
```

```
$ OMP_NUM_THREADS=$t OMP_PLACES=cores \
  OMP_PROC_BIND=close \
  srun --cpu-freq=[2000000-2000000:]performance
./a.out
```

- Perfect scaling with fixed frequency!
- Degraded scaling with Turbo Mode starting @19 cores → clock speed reduction!

Assignment 6 – Task 4

Resource-driven modeling

$$\boldsymbol{T_{\text{exec}} = \frac{W}{p_{\text{E}}}} \qquad \boldsymbol{T_{\text{data}} = \frac{V \times f}{b_S}}$$

a) No overlap

$$T_{\text{nOL}} = (T_{\text{exec}} + T_{\text{data}}) = \frac{W}{p_{\text{E}}} + \frac{V \times f}{b_S}$$

- $\boldsymbol{V}$ bytes/it. of data traffic
- $\boldsymbol{W}$ instr./it. of processor work

- $\boldsymbol{b_S}$ memory bandwidth in byte/s
- $\boldsymbol{p_{\text{E}}}$ instruction throughput in instr./cy
- $\boldsymbol{f}$ clock frequency

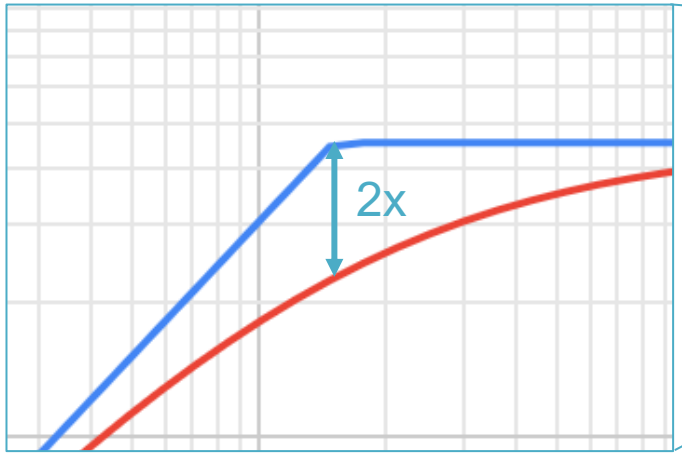
b) Full overlap

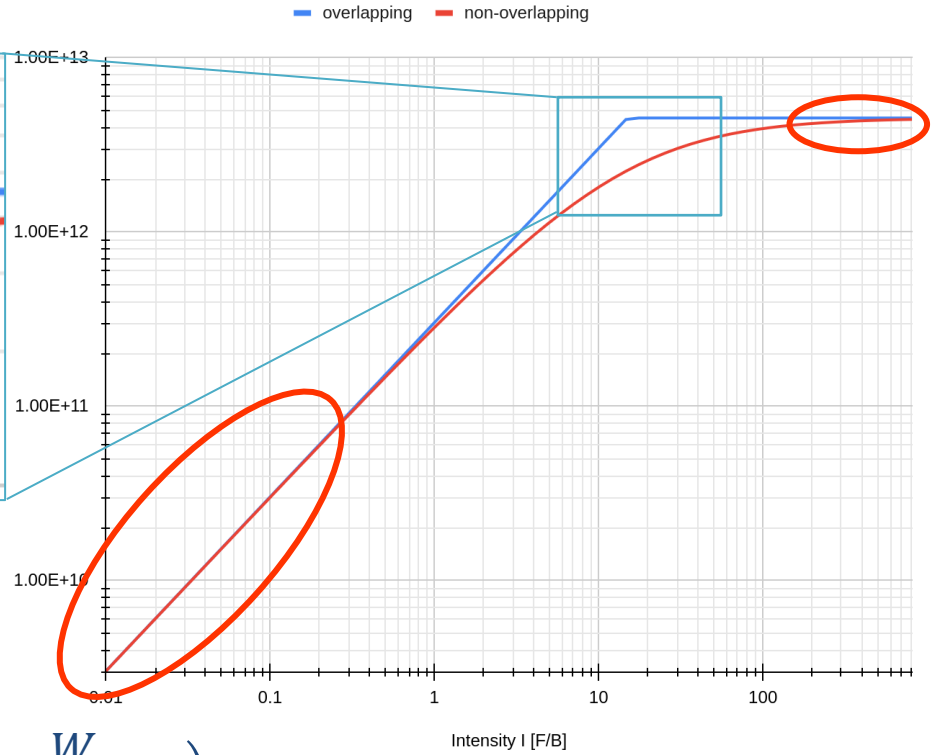
$$T_{\text{OL}} = \max(T_{\text{exec}}, T_{\text{data}}) = \max\left(\frac{W}{p_{\text{E}}}, \frac{V \times f}{b_S}\right)$$

Assignment 6 – Task 4

Resource-driven modeling

$$P = \frac{W \times f}{T}$$





$$\rightarrow P_{\mathrm{nOL}} = \frac{W \times f}{\frac{W}{p_{\mathrm{E}}} + \frac{V \times f}{b_S}} = \left(\frac{1}{p_{\mathrm{E}} \times f} + \frac{V}{W \times b_S}\right)^{-1}$$

$$\rightarrow P_{\mathrm{OL}} = \frac{W \times f}{\max(T_{\mathrm{exec}} + T_{\mathrm{data}})} = \min\left(p_E \times f, \quad \frac{W}{V} \times b_S\right)$$